ORC 67-12
FEBRUARY 1967

THE SYMMETRIC ASSIGNMENT PROBLEM

by

Katta G. Murty

OPERATIONS RESEARCH CENTER

COLLEGE OF ENGINEERING

UNIVERSITY OF CALIFORNIA - BERKELEY

THE SYMMETRIC ASSIGNMENT PROBLEM

by

Katta G. Murty
Operations Research Center
University of California, Berkeley

February 1967 ORC 67-12

This research has been partially supported by the Office of Naval Research under Contract Nonr-222-(83) and the National Science Foundation under Grant GP-4593 with the University of California. Reproduction in whole or in part is permitted for any purpose of the United States Government.

ABSTRACT

A branch and bound algorithm for finding the minimal cost symmetric assignment is discussed. The matching problem in graph theory and the Chinese postman puzzle are all special cases of the symmetric assignment problem, and hence this algorithm can be applied to solve them.

<u>Introduction</u>: The well known assignment problem is to

$$\text{minimize } z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

$$\text{subject to } \sum_{i=1}^{n} x_{ij} = 1, \text{ for } j = 1, \ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ for } i = 1, \ldots, n \qquad (1)$$

$$x_{ij} \geqq 0$$

where n is a given positive integer and $C = (c_{ij})$ is a given $n \times n$ cost matrix. C^T denotes the transpose of C. Any feasible solution to the assignment problem can be represented by the matrix $X = (x_{ij})$ where the x_{ij} satisfy conditions (1).

Consider a matrix $X = (x_{ij})$ of order $n \times n$, in which only one of the elements in each row and column is equal to 1, while all the remaining elements are zero. Such a matrix is a permutation matrix and is called an <u>assignment</u>. Every assignment is a basic feasible solution of the solution set of (1) and vice versa. We use the letters a, b to denote assignments.

Occasionally it is convenient to denote an assignment by the set of its unit cells, i.e., the cells in the matrix X representing the assignment which have unit entries in them. All the other cells have zero entries in them of course. Thus,

$$a = \{(i, j_1), \ldots, (n, j_n)\} \qquad (2)$$

where $j_1, \ldots, j_n$ is a permutation of the numbers $1, 2, \ldots, n$ is an assignment. Correspondingly we shall write $(i, j) \in a$ or $(i, j) \notin a$ to indicate that in the

matrix X representing the assignment a , the entry in the cell (i, j) is one or zero respectively.

Let $Z_C(a)$ denote the cost corresponding to assignment a , w.r.t. the cost matrix C . Hence for the assignment in (2)

$$Z_C\ (a) = \sum_{r=1}^{n} c_{rj_r}$$

when there is no ambiguity about the cost matrix, or when we are referring to the original cost matrix in (1) we will ignore the subscript and write $Z(a)$ instead of $Z_C(a)$.

If a is the assignment in (2)

$$\tilde{a} = \{(j_1, 1), \ldots, (j_n, n)\}$$

is another assignment and it is known as the reflection of a .

An assignment b is called a symmetric assignment (SA in abbreviation) if $b = \tilde{b}$, i.e., whenever it contains a cell (i, j) it should also contain the cell (j, i) . Let K denote the set of all SA. The symmetric assignment problem is the problem of finding the minimal cost (optimal) SA.

The special case of the symmetric assignment problem in which the diagonal cells bear infinite cost is equivalent to the optimal matching problem in graph theory [5]. The famous Chinese postman puzzle can be formulated as a matching problem and hence as a special case of the symmetric assignment problem (pointed out by Professor D. Gale).

Terminology:

Diagonal Cell: Any cell along the principle diagonal, i.e., a cell of the form (i, i), is called a diagonal cell.

Node: A node is a subset of K of the following form

$$\text{node } N = \{\text{all SA which contain } (i_1, j_1)(j_1, i_1) \ldots (i_r, j_r)(j_r, i_r) \text{ and which do not contain } (m_1, p_1)(p_1, m_1) \ldots (m_s, p_s)(p_s, m_s)\} . \tag{3}$$

Then the cells $(i_1, j_1)(j_1, i_1) \ldots (i_r, j_r)(j_r, i_r)$ are said to be the cells specified to be contained in the node, and the cells $(m_1, p_1)(p_1, m_1) \ldots (m_s, p_s)$ (p_s, m_s) are said to be the cells specified to be excluded from the node. The letters M, N will be used to denote the nodes.

For simplicity we can write down the node N in equation (3) as

$$N = \{(i_1, j_1)(j_1, i_1) \ldots (i_r, j_r)(j_r, i_r); \overline{(m_1, p_1)}\,\overline{(p_1, m_1)} \ldots \overline{(m_s, p_s)}\,\overline{(p_s, m_s)}\}$$

the bar above a cell indicating that it is specified to be excluded from the node. An admissible unspecified cell at node N is any cell which is unspecified at node N and which does not lie in a row or column of a cell specified to be in that node.

Branching from node N with the admissible unspecified cell (i, j). Suppose node N is given by (3) and let (i, j) be an admissible unspecified cell at N. Then neither i nor j equal any of $i_1, \ldots, i_r, j_1, \ldots, j_r$. Then we can partition N as

$$N = N_1 \cup N_2, \qquad N_1, N_2 \text{ disjoint, where}$$

$$N_1 = \{(i_1, j_1)(j_1, i_1) \ldots (i_r, j_r)(j_r, i_r)(i, j)(j, i); \overline{(m_1, p_1)}\,\overline{(p_1, m_1)} \ldots \overline{(m_s, p_s)}\,\overline{(p_s, m_s)}\}$$

$$N_2 = \{(i_1, j_1)(j_1, i_1) \ldots (i_r, j_r)(j_r, i_r); \overline{(m_1, p_1)}\,\overline{(p_1, m_1)} \ldots \overline{(m_s, p_s)}\,\overline{(p_s, m_s)}\,\overline{(i, j)}\,\overline{(j, i)}\}.$$

This operation of partitioning into two disjoint subsets is known as branching from node N with the admissible unspecified cell (i, j). The nodes N_1 and N_2 will be called the branches emanating from node N.

Reducing the matrix C, method 1: This operation proceeds as follows

(1) Subtract from each element the minimal element in its row.

(2) In the resulting matrix, subtract from each element the minimal element in its column. Suppose this leads to a matrix C_1, then all the elements in C_1 are nonnegative and each row and column of C_1 contains at least one zero element.

(3) If C_1 is not symmetric, let $C_1' = \frac{C_1 + C_1^T}{2}$. Then C_1' is a symmetric matrix with all nonnegative elements.

(4) Suppose C_1' contains some rows with no zero element at all. Pick one of them, say row i_1. By symmetry column i_1 also has no zero element.

(5) Subtract the minimum element in row i_1 from each element in that row.

(6) In the resulting matrix subtract the minimal element in column i_1 from each element in that column.

Since the operation in (5) alters only the diagonal element (i_1, i_1) in column i_1, at the end of step (6), either a pair of symmetrically placed zero cells, say (i_1, j) and (j, i_1) are created in row i_1 and column i_1 respectively or the diagonal cell (i_1, i_1) becomes a zero cell. Suppose the result is C_2.

If C_2 is not symmetric let $C_2' = \frac{C_2 + C_2^T}{2}$. By symmetry C_2' contains all the zero cells of C_1' and at least one additional zero cell in each of row i_1 and column i_1 . If C_2' has some rows and columns without any zero entries repeat steps (4), (5), (6) with C_2' replacing C_1' .

Repeat this process until finally a symmetric matrix C_R is obtained which consists of all nonnegative elements such that each row and column of C_R contains at least one zero element.

C_R is known as the reduced matrix obtained from C . The sum of all the numbers subtracted from the rows and columns during the various steps is known as the reduction of the matrix C .

Reducing the matrix C , method 2: Using this method the operation of reducing C proceeds as follows.

(1) Using the cost matrix C find the optimal assignment by the Hungarian method [1].

The Hungarian method transforms C by adding constants to its rows and columns until finally a matrix C_1 is obtained which consists of all nonnegative elements, with at least one zero in each row and column. The optimal assignment is contained among the zero cells of C_1 .

(2) If C_1 is not symmetric, then repeat step (1) with $C_1' = \frac{C_1 + C_1^T}{2}$ in place of C .

Repeat this process as many times as necessary until the final transformed matrix C_R with nonegative elements and containing at least one zero in each row and column, is symmetric.

The sum of the costs of all the optimal assignments as they are obtained in the successive steps, in known as the <u>reduction of the matrix C</u>. The final transformed matrix C_R is known as the <u>reduced matrix</u> obtained from C. It can be seen that the operation of reducing C by method 2 might involve some more work than that by method 1. However, the reduction of C obtained by method 2 is likely to be larger than that obtained under method 1 and this helps in improving the efficiency of the algorithm.

It has been proved in theorem 6 that the repetition of steps (1) and (2) of method 2 have to be carried out only a small number of times before obtaining C_R. Hence C_R is obtained from C in a small number of steps.

<u>The remaining cost matrix at node N</u>: Suppose node N is given by equation (3). Then the matrix obtained by striking off the rows and columns $i_1, \ldots, i_r, j_1, \ldots, j_r$ from C and replacing the cost elements in the cells $(m_1, p_1)(p_1, m_1) \ldots (m_s, p_s)(p_s\ m_s)$ by infinity (or a very very large positive number) is known as the <u>remaining cost matrix</u> of node N. The reduced matrix obtained from this matrix is known as the <u>reduced remaining cost matrix</u> at node N and is denoted by $C_{N,R}$.

The remaining cost matrix of K is C itself, and the reduced remaining cost matrix of K is C_R.

<u>The evaluation of an admissible unspecified cell at a node</u>. Let (i, j) be an admissible unspecified cell at a node N. Then its <u>evaluation at node N</u> is defined to be $\theta_N(i, j)$ where

$\theta_N(i, i)$ = Sum of the minimal elements in row i and column i of $C_{N,R}$ after excluding the element in (i, i)

θ_N (i, j) = Sum of the minimal elements in rows i and j and columns i and
$i \neq j$ j in $C_{N,R}$ after excluding the (i, j)th and (j, i)th elements; if these minima occur at distinct places

or = the diagonal element in (i, i) plus the sum of the minimal elements in row j and column j in $C_{N,R}$ after excluding the (i, j)th and (j, i)th elements; if minimum in row i and column i after the exclusion occurs at the diagonal cell (i, i) and the diagonal element at (j, j) is not the minimal in row j and column j after the exclusion

or = the sum of the diagonal elements (i, i) and (j, j) in $C_{N,R}$ if those elements are the minimal ones in rows i, j and columns i, j after excluding the cells (i, j) and (j, i) .

Since the reduced matrix contains all nonnegative elements, $\theta_N(i, j) \geq 0$ for all admissible unspecified cells. Also since the reduced matrix has at least one zero element in each row and column, $\theta_N(i, j) = 0$ unless (i, j) is a zero cell in $C_{N,R}$.

<u>The mathematical theory</u>

Lemma 1: If n is even, the total number of SA which do not contain any diagonal cells is $n!/2^{n/2}$.

Proof: An SA which does not contain any diagonal cell consists of pairs of cells like $(i, j)(j, i)$, $i \neq j$. When n is even the total number of ways in which n objects can be paired in this manner is $n!/2^{n/2}$ which is therefore equal to the total number of SA without any diagonal cells.

Theorem 1: The total number of possible SA is

$$\sum_{\substack{r=1 \\ r \text{ odd}}}^{n} \binom{n}{r} (n - r)!/2^{(n-r)/2} \text{ , if } n \text{ is odd}$$

$$\sum_{\substack{r=0 \\ r \text{ even}}}^{n} \binom{n}{r} (n - r)!/2^{(n-r)/2} \text{ , if } n \text{ is even.}$$

Proof: This follows by using lemma 1 and the fact that if n is odd, any SA must contain an odd number of diagonal cells and if n is even, any SA contains an even number of diagonal cells.

Theorem 2: Let C_1 be the matrix obtained by adding a constant ℓ to each element is a row (or column) of C. If b is an optimal SA w.r.t. cost matrix C then it is also an optimal SA w.r.t. cost matrix C_1 and vice versa.

Proof: This is actually a restatement of a similar theorem for the general assignment problem [1]. This follows easily because

$$Z_{C_1}(a) = Z_C(a) + \ell \text{ , for all assignments } a \text{ ,}$$

and ℓ is a constant.

Theorem 3: If C is not symmetric the optimal SA w.r.t. cost matrix C is also optimal w.r.t. the cost matrix $C' = (C + C^T)/2$.

Proof: This follows from the fact that

$$Z_C(b) = Z_{C'}(b) \text{ , for all } b \epsilon K \text{ .}$$

Theorem 4: The reduction of C is a lower bound on the cost of any SA.

Proof: If $b \epsilon K$ then by theorems 2 and 3

$$Z_C(b) = \text{reduction of } C + Z_{C_R}(b) .$$

But since each element in C_R is ≥ 0 we have $Z_{C_R}(b) \geq 0$. QED

In general, corresponding to any node N let

LB(N) = original cost elements of all the cells specified to be contained in the node + reduction of the remaining cost matrix at node N . (4)

Then from theorem 4 we see that LB(N) is a lower bound on the cost of any SA in N . We have the following theorem on the lower bounds of the branches after branching from a node.

Theorem 5: Let (i, j) be an admissible unspecified cell at node N , which is a zero cell in $C_{N,R}$. Let N_1, N_2 be the branches when N is partitioned w.r.t (i, j) .

$$N_1 = \{b \epsilon N : (i, j) \epsilon b, (j, i) \epsilon b\}$$

$$N_2 = \{b \epsilon N : (i, j) \not\epsilon b, (j, i) \not\epsilon b\} .$$

Then (i) $LB(N_2) \geq LB(N) + \theta_N(i, j)$

(ii) Let C_{N_1} be the matrix obtained by striking off the rows i, j and columns j, i from $C_{N,R}$ and let C_{N_2} be the matrix obtained by replacing the entries in the cells (i, j) and (j, i) of $C_{N,R}$ by infinity (or a very large positive number). Then

$$LB(N_1) = LB(N) + \text{reduction of } C_{N_1}$$

$$LB(N_2) = LB(N) + \text{reduction of } C_{N_2}$$

Proof: (i) This follows from the fact that by definition $b \in N_2 \Rightarrow b \in N$ and $(i, j) \notin b$, $(j, i) \notin b$.

(ii) These follow the fact that (i, j) which is admissible unspecified at N is a zero cell in $C_{N,R}$.

Theorem 6: In reducing any matrix C by method 2, step 2 need not be applied more than twice.

Proof: Suppose C_1 is the transformed matrix obtained when step 1 is applied to C. If C_1 is symmetric we are done. Otherwise we find $C_1' = \frac{C_1 + C_1^T}{2}$ which is a symmetric matrix with nonnegative elements.

Let a be an optimal assignment w.r.t. cost matrix C_1'. Let C_2 be the transformed matrix obtained when step (1) of method 2 is applied to C_1'. Then for any assignment b

$$Z_{C_1'}(b) = Z_{C_1'}(a) + Z_{C_2}(b) \qquad \text{(by the Hungarian method)}$$

But $\quad Z_{C_1'}(\tilde{a}) = Z_{C_1'}(a) \quad$ because C_1' is symmetric.

Hence $\quad Z_{C_2}(\tilde{a}) = Z_{C_2}(a) = 0$.

Since C_2 has all nonnegative elements this implies that the entries in the matrix C_2 in the cells of a and $\tilde{a}$ are all zero. This

implies that in $C_2' = \frac{C_2 + C_2^T}{2}$ there exists at least one zero cell in each row and column (corresponding to the cells of α and $\tilde{\alpha}$) and hence $C_2' = C_R$.

Theorem 7: Let N be any node. Let α be the minimal cost assignment in N .

(i) If N contains only a single assignment, then it must be α and $Z(\alpha) = LB(N)$.

(ii) If method 2 is employed for reducing matrices then $Z(\alpha) = LB(N)$.

Proof: (i) This follows from the way LB(N) is defined in equation (4).

(ii)
$$Z(b) = LB(N) + Z_{C_{N,R}}(b)$$

for all assignments b in N , by theorem 5 and

$$Z_{C_{N,R}}(\alpha) = 0 \qquad \text{(by Hungarian method)}$$

Therefore $Z(\alpha) = LB(N)$.

Algorithm:

Stage 1: Find LB(K) . If this is found by using method 2 for reducing C , the algorithm terminates if there exists an optimal assignment which is symmetric. Otherwise branch from K .

General stage m: At this stage K has been partitioned into several nodes. Any node which has not been branched is called a terminal node at this stage. In the course of the algorithm a lower bound on the cost of the minimal SA in each node (corresponding to equation (4)) has been obtained. A terminal node which has

the least value of the lower bound among all the terminal nodes is known as a minimal terminal node at this stage.

Optimality criterion: The algorithm is terminated whenever there exists a minimal terminal node which contains only one SA or when there exists a minimal terminal node in which the optimal assignment is an SA (if method 2 is used for reducing matrices).

If the optimality criterion is not satisfied at this stage, then

(i) find out the minimal terminal node with the least cardinality (i.e., which has the maximum number of cells specified to be contained in it, among all the minimal terminal nodes at this stage). Suppose it is node N

(ii) find (i, j), an admissible unspecified cell at node N such that

> (a) (i, j) is a zero cell in $C_{N,R}$ and
>
> (b) $\theta_N(i, j)$ is maximum among all the cells satisfying (a).

(iii) branch from N w.r.t. (i, j) and find the lower bounds of both the branches using theorem 5.

(iv) go to stage $(m + 1)$.

These computations are repeated until the optimality criterion is satisfied. The optimal assignment (or the only assignment) in the minimal terminal node at the final stage, which is an SA, is an optimal SA.

Justification for the algorithm: At each stage the terminal nodes are mutually disjoint and their union is K. So the optimal SA has to lie in one of the terminal nodes and it is more likely to lie in a minimal terminal node than in any other. This is the reason for branching from a minimal terminal node at each stage.

Also if the optimal assignment (or the only assignment) in a minimal terminal node is an SA then that SA must be an optimal SA by theorem 7. This proves the validity of the optimality criterion.

It can be seen that the efficiency of the algorithm improves if we could guarantee that we reach rapidly a minimal terminal node which contains a single SA. When a node N is branched as in theorem 5 it can be seen that the cardinality of N_2 is far greater than that of N_1. Hence we choose the cell (i, j) for branching N, in such a way that the lower bound of N_2 is made as large as possible. By (i) of theorem 5, the $\theta_N(i, j)$ can be used to achieve this propose and this is what we used in step (ii) of the general stage of the algorithm. The reason for branching from the minimal terminal node with least cardinality is also to help in reaching a minimal terminal node with a single SA rapidly.

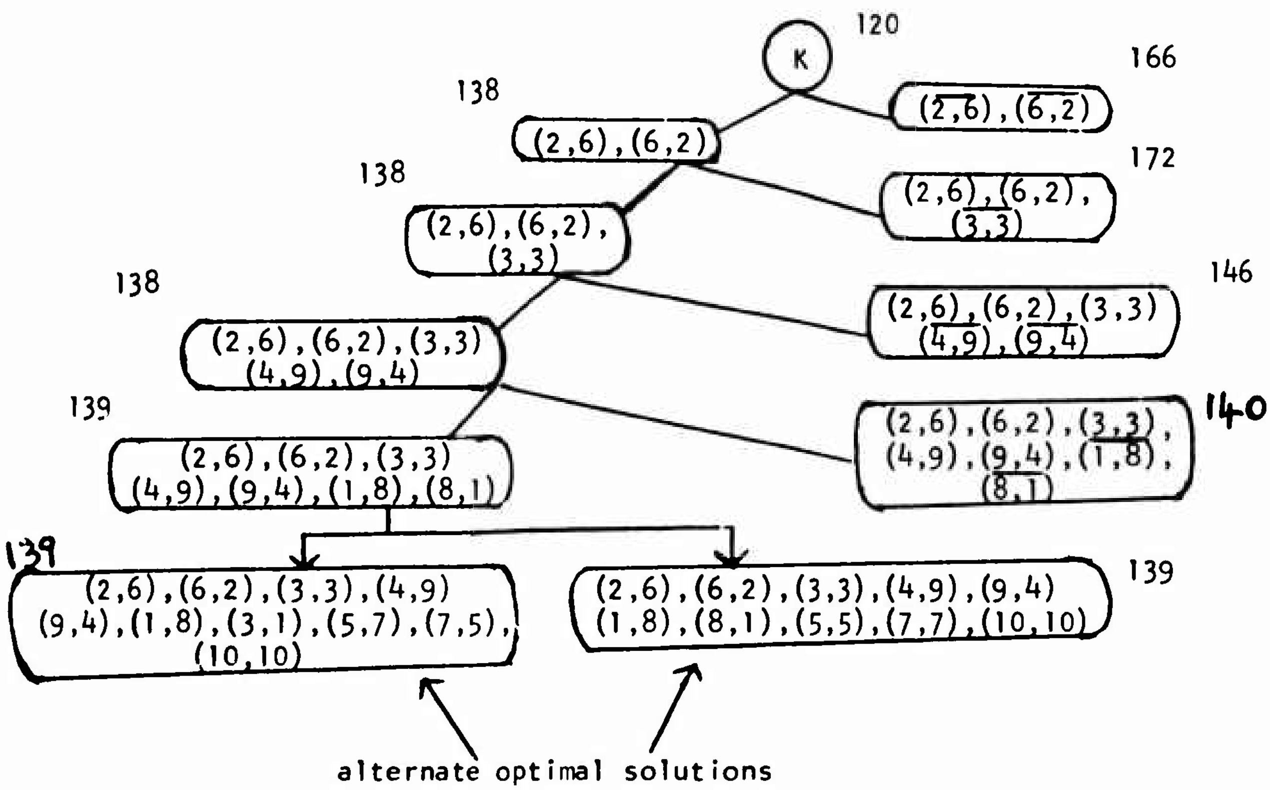
Finally method 2 for reducing matrices helps in obtaining a more precise lower bound on the cost of the minimal SA in any node. And it helps in checking whether the optimality criterion is satisfied at a much earlier stage. Thus eventhough method 2 might involve more computatinal work at each node, it may require much less branching before the optimal SA is found. Thus the use of method 2 for reducing matrices might lead to a more efficient algorithm, especially when n is large.

A numerical example:

C =

	1	2	3	4	5	6	7	8	9	10
1	10	9	73	25	3	76	52	1	35	7
2	9	63	87	79	29	3	6	11	80	94
3	73	87	1	54	68	74	32	44	44	82
4	25	79	54	41	84	98	45	47	46	85
5	33	29	68	84	5	23	12	11	32	49
6	76	3	74	98	23	78	52	98	96	56
7	52	6	32	45	12	52	19	39	64	92
8	1	11	44	47	11	98	39	3	98	27
9	35	80	44	46	32	96	64	98	59	91
10	7	94	82	85	49	65	92	27	91	14

Method 1 has been used to reduce matrices. Lower bound corresponding to each node is indicated by its side.



The chinese postman problem:

This is the problem of finding the minimal distance route passing through each edge of a connected undirected graph and then returning to the origin. Hence such a route is also called an Edge Covering Tour (E.C.T. in abbreviation). It is assumed that the distance associated with each edge of the graph is nonnegative. We denote the graph by G. We shall call the nodes of the graph as vertices to avoid confusion with the nodes defined in the branch and bound algorithm. A vertex of the graph at which an odd number of edges are incident is known as an odd vertex. If there are no odd vertices then there exists an E.C.T. in which each edge of the graph is travelled exactly once. Such an E.C.T. is known as an Euler cycle. Obviously if an Euler cycle exists it gives the minimal distance E.C.T.

If the graph contains some odd vertices then their total number is even. Let them be numbered $1, 2, \ldots, 2n$.

Lemma 2: There exists a minimal E.C.T. in which no edge of the graph G will be travelled more than twice.

Proof: Let t denote the minimal E.C.T. Obtain a new graph G' by drawing each edge of G as many times as it is travelled in t. Then t is an Euler cycle of G'.

Hence each odd node of G has become an even node in G'. Therefore, the number of repeated edges in G' incident at each odd node in G is odd, and the number of repeated edges in G' incident at each even node in G is even.

Now obtain a new graph G'' by drawing each edge of G once if it occurs an odd number of times in G' and twice if it occurs an even number of time in G'.

By the above property it is clear that all nodes are even nodes in G''. So [illegible] has an Euler cycle t'' which is an E.C.T. of G, and the total distance travelled in t'' is less than or equal to that in t. Also in t'' each edge of G is travelled either once or twice.

<u>Lemma 3</u>: There exists an optimal E.C.T., t_0, on the graph G with the following property. No edge of G is travelled more than twice. Let Γ_{t_0} be the set of edges of G which are travelled twice in t_0. Then the set of odd vertices of G can be partitioned into pairs $(i_{11}, i_{12}) \dots (i_{n1}, i_{n2})$ such that the edges in Γ_{t_0} form n different edge disjoint paths connecting each pair of odd vertices above.

<u>Proof</u>: Let G_{t_0} be the graph obtained by adding the edges of Γ_{t_0} to G. Then t_0 is an Euler cycle in G_{t_0}. Hence if i is an odd vertex in G, then Γ_{t_0} must contain a path from i to some other odd node j. Suppose along this path in Γ_{t_0} there lie two other odd vertices r, s. Adding all the edges of this path to G leaves r, s odd still. Hence by the assumptions in the hypothesis Γ_{t_0} must consist of another path between r and s. But then the E.C.T., t_1, obtained by deleting all the edges along both these paths in Γ_{t_0} between r and s is likely to be better than t_0.

Hence we can assume that there exists an optimal E.C.T., t_0, with the property mentioned in the lemma. Also in any of the paths in Γ_{t_0} between a pair of odd nodes, not more than one other odd node can lie.

<u>Lemma 4</u>: Let t_0 be an optimal E.C.T. with the property mentioned in lemma 3. Let $(i_{11}, i_{12}) \dots (i_{n1}, i_{n2})$ be a corresponding pairing of odd vertices. Then the paths in Γ_{t_0} between any pair (i_{r1}, i_{r2}) must be a shortest route between that pair (i_{r1}, i_{r2}) on G.

<u>Proof</u>: The result follows obviously because if the path in Γ_{t_0} between the pair of odd vertices were not a shortest route in G between that pair, then a better E.C.T. can be obtained by deleting the existing path in Γ_{t_0} and replacing it by the shortest route.

Lemma 2, 3, 4 together imply that the optimal E.C.T. corresponds to a minimal cost SA with the matrix of shortest route distances between odd vertices as the cost matrix. [Pointed out by Professor D. Gale.]

<u>Algorithm for the chinese postman puzzle</u>:

Let $c_{ii} = \alpha$, a very large positive number

$c_{ij} =$ total distance of the shortest route on
$i \neq j$ graph G from odd vertex i to odd vertex j ,

$i, j = 1, 2, \ldots, 2n$.

$C = (c_{ij})$.

Find the minimal SA corresponding to the cost matrix C . If the minimal SA is

$$(i_1, j_1)\ (j_1, i_1)\ \ldots\ (i_n, j_n)\ (j_n, i_n)$$

then obtain a new graph by duplicating all the edges along the shortest routes of $(i_1, j_1)\ (i_2, j_2)\ \ldots\ (i_n, j_n)$ respectively. The new graph obtained has an Euler cycle, which gives the optimal E.C.T. on G .

The proof of the algorithm follows from lemmas 2, 3, 4.

REFERENCES

[1] Kuhn, H. W., "The Hungarian Method for Solving the Assignment Problem," Naval Research Logistics Quarterly, Vol. 2, pp. 83-97, (1955).

[2] Berge, C., THE THEORY OF GRAPHS AND ITS APPLICATIONS, Methuen and Company, (1962).

[3] Ko, Kwanmei, "Graphic Programming Using Odd or Even Points," Chinese Mathematics, Vol. 1, No. 3, (1962).

[4] Edmonds, J., "Paths, Trees and Flowers," Canadian Journal of Mathematics, Vol. 17, pp. 449-467, (1956).

[5] Edmonds, J., "Maximum Matching and a Polyhedron with 0, 1 - Vertices," National Bureau of Standards, (1963).

[6] Murty, Katta, G., Caroline Karel, and John D. C. Little, "The Travelling Salesman Problem: Solution by a Method of Ranking Assignments," Case Institute of Technology, (unpublished), (1962).

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R&D (Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)		
1 ORIGINATING ACTIVITY (Corporate author) University of California, Berkeley	2a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	2b GROUP ----------
3 REPORT TITLE "The Symmetric Assignment Problem"		
4 DESCRIPTIVE NOTES (Type of report and inclusive dates) RESEARCH REPORT DATE February 1967		
5 AUTHOR(S) (Last name, first name, initial) Katta G. Murty		
6 REPORT DATE February 1967	7a TOTAL NO OF PAGES 17	7b NO OF REFS 6
8a CONTRACT OR GRANT NO ONR NONR-222(83) b PROJECT NO NR 047 033 c Res. Proj.No: RR003-07-01 d	9a ORIGINATOR'S REPORT NUMBER(S) O.R.C. 67-12 9b OTHER REPORT NO(S) (Any other numbers that may be assigned this report) none	
10 AVAILABILITY/LIMITATION NOTICES DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED		
11 SUPPLEMENTARY NOTES "Also supported by the Nat'l. Sci. Found. under Grant GP-4593"	12 SPONSORING MILITARY ACTIVITY MATHEMATICAL SCIENCE DIVISION	

13 ABSTRACT

A branch and bound algorithm for finding the minimal cost symmetric assignment is discussed. The matching problem in graph theory and the Chinese Postman puzzle are all special cases of the symmetric assignment problem, and hence this algorithm can be applied to solve them.

DD FORM 1 JAN 64 1473

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
symmetric assignment						
matching on a graph						
edge covering tool						
Chinese Postman puzzle						
branch and bound						